SPECIFICATION

COMPUTER SYSTEM, SERVERS CONSTITUTING THE SAME, AND JOB EXECUTION CONTROL METHOD AND PROGRAM THEREFOR

TECHNICAL FIELD

The present invention relates to grid computing and, in particular, to a scheduling method for assigning jobs to computers and a system configuration that implements the method.

BACKGROUND ART

A system called grid computing has attracted people's attention in recent years that integrates and uses heterogeneous information systems interconnected through a network. In this system, computer resources such as CPU power and data storage of multiple computers on the network are shared so that the system is used as a single, virtual high-performance computer. By causing the multiple computers to perform parallel processing, a large amount of processing can be performed at high speeds even though the performances of the individual computers are low.

In grid computing, scheduling which determines the execution sequence of jobs (units of processing in a program) provided to the system according to the characteristics and priorities of the jobs is extremely important. This is because, in a grid environment which is inherently heterogeneous, jobs must be assigned to multiple computers having different performances and execution of various jobs of different sizes provided from remote users as well as local users must be properly scheduled.

Scheduling schemes in grid computing can be broadly classified into two types: push and pull. In the push type scheduling (see Non-patent document 1 for example), when a job is submitted to a scheduler, the scheduler assigns a computer to execute the job and requests the computer to execute the job. For doing this, the scheduler monitors the status of the computers and assigns the job to the most suitable computer according to load information of the job.

The push type scheduling scheme is often used in a grid system called a cluster grid that consists of several hundred servers (computers) provided at the same site. The push type scheduling scheme can assign a job specifically to the most suitable computer and therefore can provide optimum scheduling. Especially, in an environment where there are

variations in behavior of computers and a job can be canceled on a computer when a user uses the computer, this scheme can provide optimum scheduling by taking such an operating environment into consideration to enable very efficient utilization of the system.

In the pull type scheduling (see Non-patent document 2 for example) on the other hand, each computer polls a scheduler to request a job when the computer becomes ready to execute a job. If there is a job to execute when the request is made, the scheduler assigns the job to the computer and the computer executes the assigned job. If there is no job to execute, the computer polls again after a lapse of predetermined time.

The pull type scheduling scheme can be implemented with a very simple configuration and, therefore, it is often used in grid systems consisting of several thousand computers. Especially, a grid built on the Internet uses this pull scheduling scheme because of constraints of the network. Furthermore, the pull type scheduling scheme can be readily applied to a large number of computers because management information about the computers is simple. Self-optimization by polling (that is, computers with higher availability perform polling more frequently) can improve the efficiency to some extent.

Non-patent document 1: Chris Smith, "Open Source Metascheduling for Virtual Organizations with the Community Scheduler Framework (CSF)", Technical Whitepaper, Platform Computing Inc., August 2003

Non-patent document 2: Eric Korpela, Dan Werthimer, David Anderson, Jeff Cobb, Matt Lebofsky, "Massively Distributed Computing for SETI", Computing in Science & Engineering, Vol. 3, Issue 1, Jan.–Feb. 2001, Pages 78–83

DISCLOSURE OF THE INVENTION

PROBLEMS TO BE SOLVED BY THE INVENTION

As described above, the conventional push type scheduling scheme in grid computing enables optimum job assignment. However, in order to accomplish optimum scheduling, the utilization of each computer must be managed. Accordingly, operations of scheduler are complicated and changes to a system configuration (such as addition or removal of a compute that executes jobs) cannot readily be accommodated. Furthermore, the push type scheduling cannot be applied to computers inside a firewall because the scheduler accesses each computer to request it to execute a job.

On the other hand, the pull type scheduling scheme can readily accommodate changes to a system configuration because computer management information required by the scheduler is simple. Furthermore, jobs can be assigned to computers inside a firewall

because access on the network is performed by polling from the computers. Although the pull type scheduling scheme enables self-optimization by the polling, it causes a time loss. That is, even when there is a job to execute, a request for execution of the job is issued only after polling from a computer. Moreover, when a computer performs polling, a job that is optimum to the polling computer is assigned to the computer among jobs to be executed, which may not result in the most efficient scheduling for the whole system because, depending on the type of the job, there can be more suitable computers.

Therefore an object of the present invention is to provide a system and a job execution control method capable of implementing optimum scheduling in grid computing based on the types and utilization status of computers as well as the types of jobs, and readily accommodating changes to a system configuration.

Another object of the present invention is to enable optimum job assignment equivalent to that of push type scheduling even in a grid including computers access to which is controlled through a firewall.

Yet another object of the present invention is to provide a scheduling method that is a combination of push type scheduling and pull type scheduling, and to provide a system using the method.

MEANS FOR SOLVING THE PROBLEMS

In order to achieve the objects, the present invention provides a computer system for performing grid computing by using multiple computers interconnected through a network and is configured as follows. The computer system includes a center server which is a computer requesting computers on the network to execute a job, and process servers each of which is a computer executing a job in response to a request from the center server. The center server includes a scheduler section which assigns a job to be executed to a process server and issues a job execution request, and an agent section which manages information about the process server, receives the request issued by the scheduler section, and sends the request to the process server to which the requested job has been assigned, depending on the status of the process server.

In particular, multiple agent sections are provided in association with the multiple process servers in one-to-one correspondence.

Preferably, the agent section obtains from a corresponding process server and manages information about the capacity and operating status of the corresponding process server. The scheduler section assigns a job to the process server on the basis of the information managed by the agent section.

The agent section sends a request received from the scheduler section in response to a polling access from the process server, or sends a request received from the scheduler section at timing managed by the agent section, depending on an access type of the process server.

In particular, if a process server is connected to the center server through a firewall and cannot be accessed by the center server across the firewall, the agent section waits for a polling access from the process server before sending a request.

In another aspect of the present invention that achieves the objects described above, a job execution control method is provided to perform scheduling of jobs in a grid computing system and request execution of the jobs, using a computer. The job execution control method includes the steps of assigning a job to a process server constituting a system and executing a job, on the basis of the capacity of the process server stored in a storage regardless of the operating status of the process server, issuing a job execution request to the process server to which the job is assigned, and holding temporarily the issued job execution request and sending the job execution request to the process server to which the job is assigned, depending on the operating status of the process server.

The present invention further provides a program for controlling a computer to implement the functions of the center server described above or for causing a computer to perform operations equivalent to the steps of the job control method described above. The program may be distributed with a magnetic disk, optical disk, semiconductor memory, or other recording medium in which the program is stored, or distributed through a network.

ADVANTAGES OF THE INVENTION

According to the present invention configured as described above, an agent section that relays communication between a scheduler and a process server in grid computing is provided on a center server and the agent section performs control to accommodate a difference in access types among process servers, thereby enabling optimum scheduling in terms of the types and utilization status of computers as well as the types of jobs regardless of the difference in access types among the process servers. Furthermore, because the agent section is provided for each process server, a system configuration can be readily modified by adding or removing a process server or servers.

According to the present invention, because the agent section performs control to accommodate a difference in the access types among the process servers, a process server capable of directly receiving a job execution request through the push type scheduling may coexist with a process sever receiving a job execution request after polling through the pull type scheduling, in the system, and optimum job assignment equivalent to that of the push

type scheduling can be accomplished.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 shows a general configuration of a grid computing system according to an embodiment;

Figure 2 schematically shows a hardware configuration of a computer suitable for implementing a center server and a process server in the grid computing system according to the embodiment;

Figure 3 shows a functional configuration of a center server according to the embodiment;

Figure 4 shows a relationship between a functional configuration of a process server that performs polling and a PS agent section according to the embodiment;

Figure 5 shows a relationship between a functional configuration of a process server that does not perform polling and a PS agent section according to the invention; and

Figure 6 is a flowchart illustrating an operation performed by a PS agent section when a job execution request is issued according to the embodiment.

BEST MODE FOR CARRYING OUT THE INVENTION

The best mode for carrying out the present invention (hereinafter referred to as the embodiment) will be described in detail below with reference to the accompanying drawings.

Figure 1 shows a general configuration of a grid computing system according to the present embodiment.

As shown in Figure 1, the system of the present embodiment includes a center server (CS) 100 which assigns jobs and process servers (PS) 200 which actually execute jobs assigned by the center server 100. The center server 100 and the process severs 200 are interconnected through the Internet or other computer network. The computer network may be a wired or wireless communication network using any communication protocol, and may have a firewall or any other access control.

Figure 2 schematically shows a hardware configuration of a computer suitable for implementing the center server 100 and the process server 200 in the grid computing system according to the present embodiment.

The computer shown in Figure 2 includes a CPU (Central Processing Unit) 11 which is processing means, a main memory 13 connected to the CPU 11 through a M/B (mother board) chip set 12 and a CPU bus, a video card 14 also connected to the CPU 11 through the M/B

chip set 12 and an AGP (Accelerated Graphics Port), a magnetic disk device (HDD) 15 and a network interface 16 both connected to the M/B chip set 12 through a PCI (Peripheral Component Interconnect) bus, and a flexible disk drive 18 and keyboard/mouse 19 both connected to the M/B chip set 12 through the PCI bus, a bridge circuit 17, and a low-speed bus such as an ISA (Industry Standard Architecture) bus.

It should be noted that the hardware configuration of the computer that implements the present embodiment shown in Figure 2 is illustrative only and any other configurations may be used as long as the present embodiment can be applied thereto. For example, instead of the video card 14, a video memory alone may be provided and image data may be processed in the CPU 11. An external storage device such as a CD-R (Compact Disc Recordable) and DVD-RAM (Digital Versatile Disc Random Access Memory) drive may be provided through an interface such as an ATA (AT Attachment) or SCSI (Small Computer System Interface).

Figure 3 shows a functional configuration of the center server 100. The center server 100 includes a scheduler section 110 which performs job assignment (scheduling) for each process server 200, and a PS agent section 120 which manages a process server 200 and relays requests and responses transmitted to and from the process server 200. The PS agent section 120 is provided for each of the process servers 200 making up the grid computing system. The scheduler section 110 accesses each process server 200 through the PS agent section 120.

The scheduler section 110 may be implemented by the program controlled CPU 11 and memory means such as the main memory 13 and the magnetic disk device 15 shown in Figure 2, for example, and includes, as specific functional units, a PS capacity checking section 111, an optimum PS selecting section 112, and a job requesting section 113 as shown in Figure 3.

The PS capacity checking section 111 makes an inquiry to the PS agent sections 120 corresponding to the respective process servers 200 about the capacity of the process servers 200 to obtain the information.

The optimum PS selecting section 112 selects an optimum process server 200 for a job on the basis of the capacity information of the process servers 200 obtained by the PS capacity checking section 111 and assigns the job to that process server 200. Any optimization logic may be used in the job assignment.

The job requesting section 113 issues a request for job execution to a PS agent section 120 corresponding to a process server 200 selected by the optimum PS selecting section 112.

In the present embodiment, the PS agent section 120 relays communication between the scheduler section 110 and the process server 200 and receives a job execution request from the scheduler section 110 on behalf of the process server 200. Accordingly, the PS capacity checking section 111 makes an inquiry to the PS agent 120 and the job requesting

section 113 issues a request to the PS agent 120. However, the function of the scheduler section 110 is the same as that of the existing push type scheduler. Therefore, a scheduler used in the existing grid computing system can be used as the scheduler section 110.

The PS agent section 120 may be implemented by the program controlled CPU 11 and memory means such as the main memory 13 and the magnetic disk device 15 shown in Figure 2, for example, and includes, as specific functional units, a PS status managing section 121, a PS capacity managing section 122, a job receiving section 123, a job requesting section 124, and a polling waiting section 125, as shown in Figure 3.

The PS status managing section 121 accesses the corresponding process server 200 to ascertain the current operating status of the process server 200.

The PS capacity managing section 122 manages statistical information concerning the job execution capacity of the process server 200 and, in response to an inquiry from the PS capacity checking section 111 of the scheduler section 110, returns information managed by the PS capacity managing section 122. The statistical information concerning the job execution capacity includes static information about the throughput of a CPU and the storage capacity of a storage device as well as information obtained by statistically processing dynamic information such as changes of load on the CPU with time and the operation pattern of the CPU.

Information managed by the PS status managing section 121 and the PS capacity managing section 122 is obtained from the process server 200 corresponding to the PS agent section 120 and is stored in storage means such as the main memory 13 or the magnetic disk device 15 shown in Figure 2.

The job receiving section 123 receives a job execution request issued by the job requesting section 113 of the scheduler section 110.

The job requesting section 124 transfers a job execution request received at the job receiving section 123 to the corresponding process server 200.

The polling waiting section 125 receives notification that the process server 200 is ready to execute a job, from the process server 200 through polling.

According to the present embodiment, while a push type scheduler is used, a polling access from a process server 200 can be accepted and a job execution request can be sent in response to the polling access, which will be detailed later. The polling waiting section 125 in the PS agent section 120 is used for receiving such polling and, therefore, is not a necessary component for the PS agent section 120 for a process server 200 capable of receiving, without polling, a job execution request sent from the center server 100 at desired timing in the center server 100.

A relationship between a functional configuration of a process server 200 and a

corresponding PS agent section 120 will be described below.

As mentioned above, some of the process servers 200 assumed in the present embodiment perform polling and the others do not perform polling.

Figure 4 shows the relationship between a functional configuration of a process server 200 that performs polling and a corresponding PS agent section 120.

The process sever 200 includes a PS embedded section 210 which allows a computer as shown in Figure 2 to function as a process server 200 in the grid computing system.

The PS embedded section 210 may be implemented by the program controlled CPU 11 and storage means such as the main memory 13 or the magnetic disk device 15 shown in Figure 2, for example, and includes, as specific functional units, a PS status monitoring section 211, a PS status notifying section 212, a job receiving section 213, a job executing section 214, and a polling section 215 as shown in Figure 4.

The PS status monitoring section 211 monitors the current utilization status of the process server 200 and status of resources to collect information.

The PS status notifying section 212 provides information about the utilization status of the processing server and the status of the resources collected by the PS status monitoring section 211 to the PS agent section 120 of the center server 100. The PS status managing section 121 and the PS capacity managing section 122 in the PS agent section 120 receive the notification and store it in a storage device such as the main memory 13 or magnetic disk device 15 for management purpose. Notification from the PS status notifying section 212 to the center server 100 may be provided at regular intervals or may be provided when the operating status of the process server 200 has changed. Alternatively, the corresponding PS agent section 120 on the center server 100 may make an inquiry to the process server 200 at any timing.

The job receiving section 213 receives a job execution request sent from the job requesting section 124 of the PS agent section 120 of the center server 100.

The job executing section 214 executes a job received at the job receiving section 213 by using resources of the process server 200.

The polling section 215 notifies the PS agent section 120 on the center server 100 that the process server 200 is ready to execute a job if the polling section 215 determines so on the basis of information monitored by the PS status monitoring section 211. The polling waiting section 125 in the PS agent section 120 receives the notification from the polling section 215 and causes the job requesting section 124 to send a job execution request.

It should be noted that the polling section 215 can be omitted from a process server 200 if the server is capable of receiving a job execution request sent from the center server 100 without performing polling, that is, if the process server 200 is not placed inside a

firewall or otherwise access controlled.

Figure 5 shows a relationship between a functional configuration of a process server 200 that does not perform polling and a PS agent section 120.

Operations of the grid computing system configured as described above according to the present embodiment will be described below.

As mentioned above, the scheduler section 110 of the center server 100 is the same as the existing push type scheduler. Therefore, when a job to be executed occurs, the scheduler section 110 tries to assign the job to any of the process servers 200 being managed by the center server 100. The scheduler section 110 obtains statistical information about the capacities and operation patterns of the process servers 200 from the PS agent sections 120 and performs optimum scheduling on the basis of the information and the type and characteristics of the job. The scheduler section 110 issues a job execution request regardless of the operating status of the process server 200 to which the job has been assigned, and sends it to the PS agent section 120 corresponding to the process server 200 to request it to execute the job.

The PS agent section 120 operates differently depending on whether the access type of the process server 200 is the one capable of receiving a job execution request directly from the center server 100, or the one receiving a job execution request after polling because the process server 200 is placed inside a firewall.

Figure 6 is a flowchart illustrating an operation of a PS agent section 120 when it receives a job execution request.

As shown in Figure 6, the PS agent section 120 receives a job execution request from the scheduler section 110 through the job receiving section 123 (step 601). If a process server 200 corresponding to the PS agent section 120 is of the type capable of directly receiving a job execution request, the PS agent section 120 imrelayly sends the job execution request to the process server 200 (steps 602 and 604).

On the other hand, if the process server 200 corresponding to the PS agent section 120 is of the type that performs polling to receive a job execution request, the PS agent section 120 waits until the process server 200 polls (steps 602 and 603), then sends the job execution request received from the scheduler 110 to the process server 200 (step 604). If the PS agent section 120 does not have a job execution request to be sent when the process server 200 polls, the PS agent section 120 performs no action and waits until a job and next polling are received.

If the process server 200 is of the type capable of directly receiving a job execution request from the center server 100, the process server 200 receives the job execution request, executes the job according to the request, and then returns the result of the execution to the PS

agent section 120 of the center server 100.

On the other hand, if the process server 200 is of the type that performs polling before receiving a job execution request, the process server 200 performs polling when it becomes ready to execute a job. Then the process server 200 waits until a job execution request is sent. When it receives a job execution request sent from a corresponding PS agent section 120 on the center server 100, the process server 200 executes the job according to the request and returns the result of the execution to the PS agent section 120 on the center server 100. If the process server 200 has not received a job execution request, it repeats polling after a lapse of predetermined time.

The operation of the process server 200 described above is the same as that of a process server in a conventional grid computing system. However, in the present embodiment, each of the PS agent sections 120 on the center server 100 corresponding to the respective process servers 200 can control whether to send a job execution request to a corresponding process server 200 at timing it manages or to wait for polling access from the process server 200 before sending a job execution request, according to the access type of the process server 200, as described above. This means that the difference in the access types among the process servers 200 can be accommodated by the control by the PS agent sections 120. Therefore, a process server 200 capable of receiving a job execution request directly from the center server 100 can coexist with a process server 200 that is placed inside a firewall and performs polling before receiving a job execution request in the system according to the present embodiment.

As described above, the configuration according to the present embodiment can include a process server 200 that performs polling before receiving a job execution request. Even in this case, the scheduler section 110 can perform optimum scheduling based on the capacity of the process server 200 as well as the type and characteristics of a job regardless of the operating status of the process server 200 because the scheduler section 110 issues the job execution request to the process server 200 through the PS agent sections 120.

Furthermore, according to the present embodiment, the PS agent sections 120, each of which corresponds to one process server 200, are provided on the center server 100, and the PS agent sections 120 manage information about the corresponding process servers 200 and control transmission and reception of requests and responses as described above. Furthermore, the scheduler section 110 assigns a job to a process server 200 on the basis of the information about the process server 200 managed by a corresponding PS agent section 120. Therefore, when a process server 200 is to be added to or removed from the system, the system configuration can be readily changed by adding or removing a PS agent section 120 corresponding to the press server.